СУПЕРКОМПЬЮТЕРЫ И ИЕРАРХИЧЕСКАЯ МОДЕЛЬ ПАРАЛЛЕЛЬНОГО ПРОГРАММИРОВАНИЯ HPGAS

В.С. Горбунов, А.В. Зайцев, Д.Л. Аверичева, Д.В. Андрюшин, Л.К. Эйсымонт

Введение

Наращивание производительности суперкомпьютеров в настоящее время производится путём увеличения количества вычислительных узлов, повышения их вычислительной мощности, увеличения объёмов памяти и скорости передачи данных по межузловой сети. Не освоенным в России направлением при построении суперкомпьютеров является использование многосокетных узлов, в которых четыре и более процессорных СБИС соединены быстрой локальной сетью (например, на базе интерфейсов HyperTransport и QPI), а также применяется многоблочная оперативная память и несколько межузловых сетевых адаптеров.

Представляется перспективным подход, когда некоторые группы узлов дополнительно объединятся ещё одной сетью, имеющей высокую скорость передачи коротких сообщений. Это позволяет гибко использовать сетевые ресурсы, подключая по необходимости такую сеть вместо стандартной, а также получая более широкий по пропускной способности межузловой канал при использовании одновременно двух сетей. Более того, появление таких межузловых сетей (например, на базе PCI-express или HyperTransport) позволило ввести ещё один уровень иерархии в виде *макроузлов* и ввести на этом уровне общую память [1].

Увеличение сложности архитектуры суперкомпьютеров, связанное с появлением многосокетных узлов, характерно не только для суперкомпьютеров кластерного типа, это можно увидеть и в заказных суперкомпьютерах, только там используются специальные решения. Однако, общая характерная черта – усложняется структура суперкомпьютеров, увеличиваются уровни иерархии памяти и коммуникаций [2]. В качестве примеров можно привести российский кластерный суперкомпьютер К100 [3], американский заказной суперкомпьютер IBM BlueWaters, китайский заказной суперкомпьютер Tianhe-1A.

В дополнение (а может, на смену) к популярной в настоящее время модели организации параллельных программ MPI существует модель организации вычислений с общими массивами данных, PGAS. Эта модель была предложена давно и предполагает наличие двух уровней иерархии памяти – локальная и общая. В настоящее время, как уже упоминалось, уровней иерархии гораздо больше. В связи с этим приобрело актуальность решение проблемы отражения реальных уровней иерархии памяти и коммуникаций в средствах программирования.

В связи с постановкой этой проблемы могут возникнуть сомнения – а надо ли это делать, поскольку это заведомо усложнит параллельное программирование? Наше мнение – надо, поскольку лучше программиста эффективно воспользоваться возможностями иерархической памяти вряд ли кто сможет. Другое дело, что эти уровни иерархии должны быть введены максимально необременительно для пользователя. Собственно говоря, в этом суть представленной далее модели HPGAS (иерархической PGAS).

Ещё один вопрос - на каких пользователей и на какие задачи может быть ориентирована модель HPGAS? Если использовать методику разделения задач на классы CF- (cache friendly, с хорошей пространственно-временной локализацией обращений к памяти) и DIS- (data intensive systems, с плохой пространственно-временной локализацией обращений к памяти), то можно заметить следующее. Задачи CF-класса оказались менее чувствительными к усложнению структуры суперкомпьютеров, и продолжают демонстрировать их высокую производительность даже при использовании стандартных средств параллельного программирования, например библиотеки MPI. Это происходит благодаря возможности заранее спланировать большую часть межузловых обменов, а также оптимальным образом разместить данные в памяти. К сожалению, такие методики, как правило, не применимы для задач класса DIS, ориентированных большей частью не на вычисления, а на работу со сверхбольшими объёмами данных, причём, когда доступ к данным носит интенсивный, нерегулярный и непредсказуемый характер. Повышение реальной производительности при выполнении задач обоих классов возможно с помощью подходов, учитывающих неодинаковое время доступа к данным, расположенным на разных уровнях иерархии памяти.

Иерархическая организация суперкомпьютера и памяти.

Уточним, как нами представляется иерархическая организация суперкомпьютеров. В качестве примера на рис.1 представлены физические уровни организации суперкомпьютера ПТК Санкт-Петербургского Государственного Политехнического Университета, который создан по типу суперкомпьютера К100, но с применением “тяжелых” 4-х сокетных SMP-узлов на 12-ядерных микропроцессорах Magny Course, общая для процессоров сокетов узловая память может достигать 512 Гбайт.

Уровни иерархии этого суперкомпьютера таковы: сокет с процессором и подключенными блоками памяти; 4- сокетная серверная плата; группа соединенных через МВС-коммутатор серверных плат; суперкомпьютер в целом, состоящий из макроузлов, соединенных сетью Infiniband. При этом важно

подчеркнуть, что серверные платы макроузлов имеют адаптеры сети Infiniband, причем даже не по одному такому адаптеру. Итак, макроузлы имеют “множественные выходы” в сеть Infiniband.

Таким образом, процессорные ядра такого иерархического суперкомпьютера имеют и иерархическую нумерацию, они адресуются пятеркой вида < Nm , Nb , Ns , Ncs, Nc > , где: Nm - номер макроузла; Nb – номер серверной платы в макроузле; Ns – номер процессорного сокета в макроузле; Ncs – номер чип-сокета в процессорном сокете; Nc – номер ядра в чип-сокете. Поясним понятия процессорного сокета и чип-сокета.

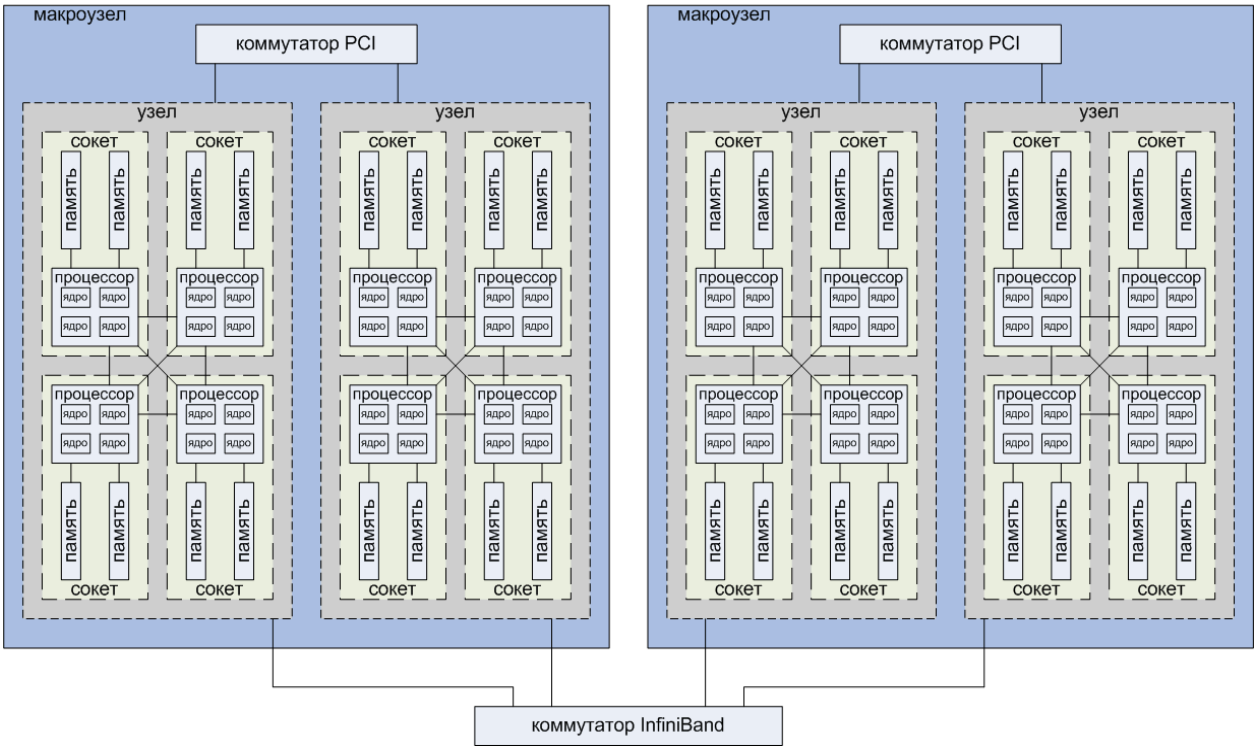

Рис. 1. Физические уровни иерархии суперкомпьютеров

Дело в том, что в серверных платах имеются сокеты, в которых могут находиться либо однокристальные микропроцессоры, либо многокристальные микропроцессоры (например, микропроцессор MagnyCourse, который состоит из двух 6-ядерных кристаллов, рис.2), либо платы с микропроцессорами (например, 2-х процессорные платы с 10-ядерными микропроцессорами Westmere-EX, рис.3). Принцип выделения в сокете чип-сокета таков – выделяем кристалл, к которому через встроенные в кристалл контроллеры памяти подключены блоки памяти. То, что находится в чип-сокете, рассматриваем как атомарный “строительный” блок суперкомпьютера.

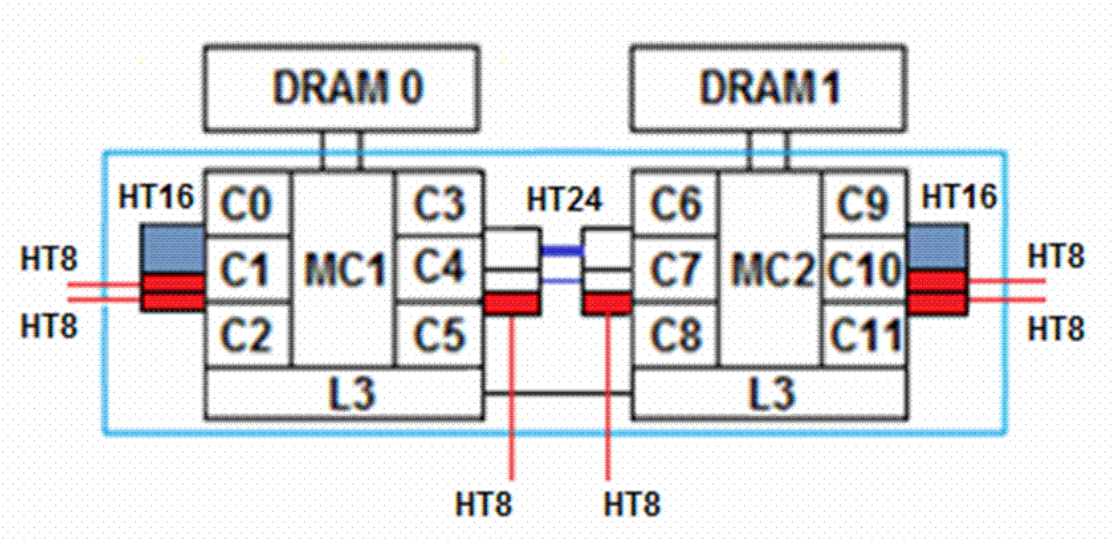

Рис. 2. 2-х кристальный микропроцессор Magny Course в процессорном сокете платы Supermicro (два чип-сокета)

Физические блоки памяти подключены к чип-сокетам через встроенные в кристаллы контроллеры памяти, причем таких контроллеров может быть несколько. Когда говорим об уровнях иерархии памяти, то при этом подразумеваем группы блоков памятей чип-сокетов, имеющих общую сеть доступа к ним. Например, если обратиться к рис.2, то на первом уровне иерархии находятся памяти двух чип-сокетов, соответственно DRAM0 и отдельно DRAM1. На втором уровне иерархии - память DRAM0 и DRAM1 вместе. Доступ к ним возможен из ядер C0-C5 одного кристалла и ядер C6-C11 другого кристалла. При этом, для выполнения обращения используется внутрипроцессорное 24-лэйновое соединение HT24. Такое соединение заметно проявляется при выполнении задач. Например, время выполнения задачи UA (из пакета NASA, класс C) на ядрах C0-C5 при работе только с DRAM0, меньше в 1.5 раза, чем время выполнения при работе только с DRAM1.

На следующем уровне иерархии находится узел, он изображен на рис.1. Все блоки памяти процессорных сокетов образуют NUMA систему общей памяти. Но доступ к различным блокам памяти осуществляется с использованием внутриузловой сети на 8-лэйновых линках HyperTransport. Такая организация может еще больше снизить скорость обращений к памяти между процессорными сокетами. Например, на упоминавшейся ранее задаче UA работа с памятью удаленного сокета может увеличить время счета в 2-3 раза, в зависимости от используемого сокета.

Следующий уровень иерархии памяти – модули узлов, доступные через сеть MBC-экспресс, это узлы макроузла. Более высокий уровень иерархии - память, объединенная сетью Infiniband. Заметим, что возможно и продолжение этих уровней иерархии, если один суперкомпьютер рассматривать как узел ГРИД-сети. Это могло бы быть полезным для реализации облачных вычислений.

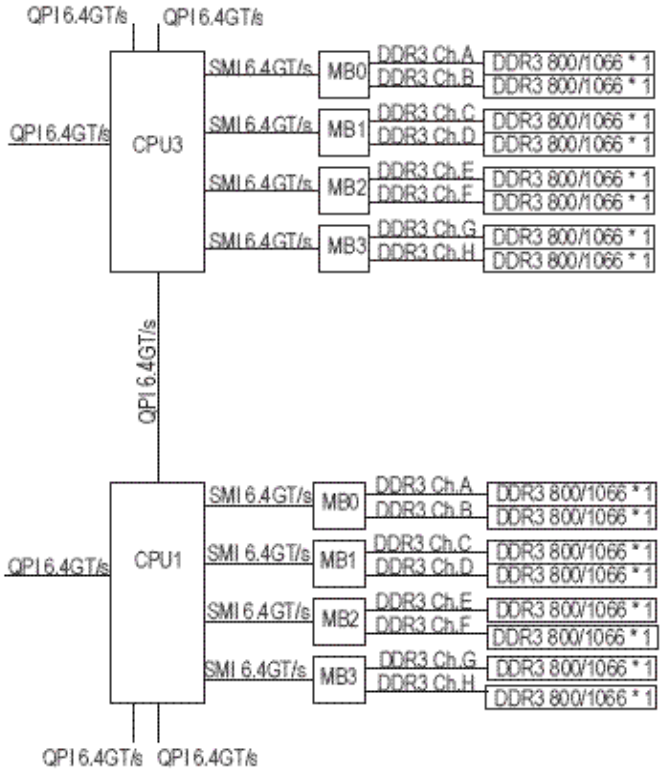

Рис. 3. 2-х процессорная плата, вставляемая в один из четырех сокетов платы Supermicro X8OBN-F (два чип-сокета).

Модель HPGAS.

Такую физическую иерархию памяти на логическом уровне можно представить так, как показано на рис.4. Вопрос в том, как отобразить такую логическую организацию на физические модули памяти, входящие в сложную иерархическую структуру блоков современного суперкомпьютера. При этом важно не сильно затруднить программирование на уровне пользователя. В этом суть предлагаемой модели HPGAS.

Базовая идея состоит в том, чтобы обобщить понятие сегмента виртуальной памяти, привязывая именно его к физической памяти с учетом ее иерархии. При этом предлагается конкретный шаг – обобщение понятия V-сегментов проекта Ангара [4]. Надо отметить, что уже V-сегменты сами по себе были в свое время обобщением того, что предлагалось ранее, но при этом была ориентация на одноуровневое множество вычислительных узлов и доступ через однородную сеть, иерархия не учитывалась. Это и отражалось в полях дескрипторов V-сегментов. В рассматриваемом случае модели HPGAS структура этих дескрипторов должна усложниться.

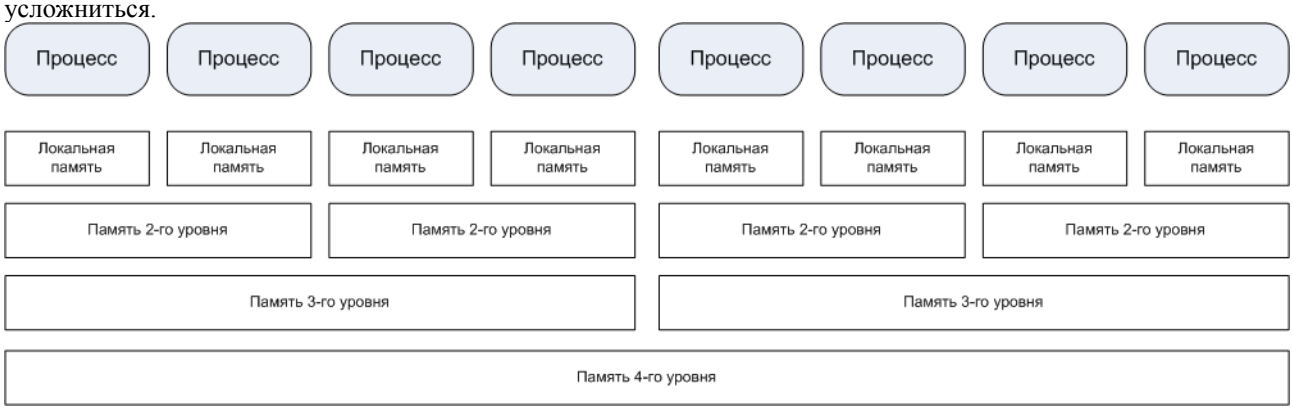

Рис. 4. Логические уровни иерархии памяти

Сегмент модели HPGAS будет содержать атрибуты, которых ранее не было и которые отражают его размещение на макроузлах, серверных платах, процессорных сокетах и чип-сокетах. Кроме этого будут атрибуты задания функции отображения линейного виртуального адресного пространства сегмента на физическую память множества заданных модулей памяти. Рассматриваются и другие атрибуты, управляющие

выполнением обращений к памяти сегмента, они могут быть взяты такими, как в проекте Ангара [4], либо немного доработаны.

Информация из дескриптора V-сегмента учитывается при трансляции виртуального адреса, по которому происходит обращение к памяти. В зависимости от того, на каком уровне определен сегмент, задействуются для доступа к его ячейкам сети того или иного уровня. Этот процесс может быть реализован программно в процессорном ядре, может быть вынесен в адаптер сети, а при разработке собственного микропроцессора – реализован в блоке трансляции виртуальных адресов и адаптере многосвязной сети.

В настоящее время рассматривается способ включения этого механизма сегментов в средства программирования, на уровне библиотеки Shmem. В этом случае номер узла является номером сегмента. На уровне языков класса PGAS (например, UPC) можно ввести дополнительную спецификацию задаваемой разделяемой памяти, которая и будет спецификацией сегмента.

Практическая работа языка программирования на базе Shmem предполагается на кластерах с сетью МВС-экспресс, а также на имитационной модели перспективного суперкомпьютера экзафлопсного класса.

ЛИТЕРАТУРА:

1. В.С. Горбунов, А.О. Лацис, А.Н. Иванов "О построении суперкомпьютеров на основе интерфейса PCI – Express." // Материалы первой конференции по суперкомпьютерам СКТ-10, октябрь 2010г.
2. В.С. Горбунов, В.К. Левин, С.В. Яблонский "Суперкомпьютеры сегодня и завтра" // Материалы первой конференции по суперкомпьютерам СКТ-10, октябрь 2010г.
3. В.С. Горбунов "Архитектура хорошо масштабируемого вычислительного кластера" // Материалы первой конференции по суперкомпьютерам СКТ-10, октябрь 2010г.
4. А.С. Семенов, А.А. Соколов, Л.К. Эйсымонт "Архитектура глобально адресуемой памяти мультитредово-потокового суперкомпьютера" //«Электроника: НТБ», №1, 2009 г., с. 50-56. http://www.electronics.ru/pdf/1_2009/1962.pdf